Angular JS

Session 2

G Jayendra Kartheek

Vishwarajsinh Sodha

Facts

- StackOverFlow survey
- Large Websites uses Angular
- Google Survey

Things discussed in session 1

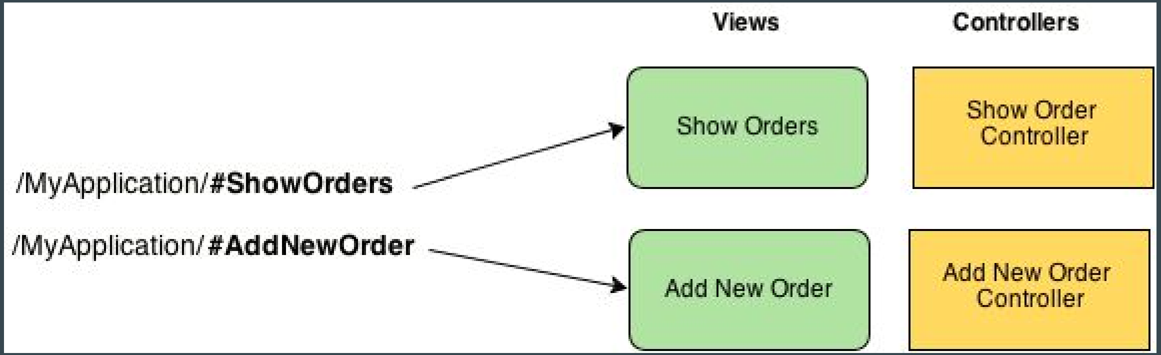
1. Data Binding
2. Scope and rootScope
3. Dependency Injection
4. Modules
5. Controllers
6. Ajax and Single Page Application

Agenda

- Routing
- Service
- Factory
- Communicating between directives ($emit, $broadcast and $on)
- Assignment

Routing

- Routing helps you in dividing your application in logical views and bind different views to Controllers



$routeProvider

- Routing in angularjs is taken care using angular inbuilt service called $routeProvider
- Dependency Injection is used to inject the routeprovider into the controller
- Methods
  - Config : method to configure $routeProvider
  - When() : define the routing page
  - otherwise() : default routing page
- Shown in the view using a custom inbuilt directive <ng-view>

Add View to the Page

You can define the ng-view in the following methods

1. <div ng-view=""></div>

2. <ng-view></ng-view>

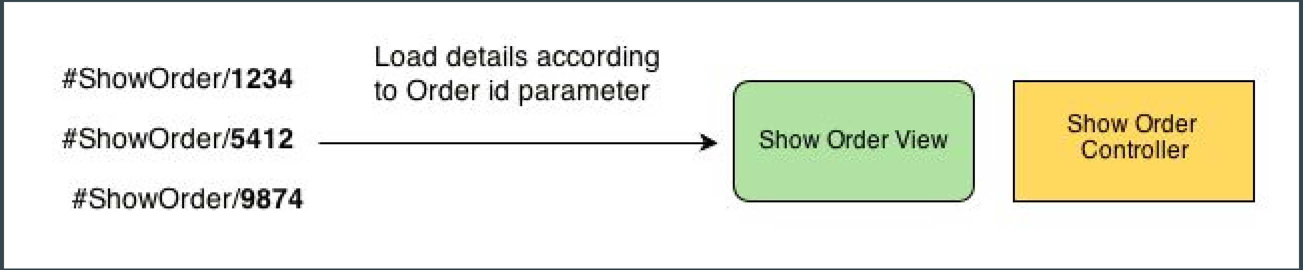
3. <div class="ng-view"></div>

Adding routing to the views

```
var sampleApp = angular.module('test', []);

sampleApp .config(['$routeProvider',
  function($routeProvider) {
    $routeProvider.
      when('/addOrder', {
        templateUrl: 'templates/add-order.html',
        controller: 'AddOrderController'
      }).
      when('/showOrders', {
        templateUrl: 'templates/show-orders.html',
        controller: 'ShowOrdersController'
      }).
      otherwise({
        redirectTo: '/addOrder'
      });
  }]);
```

Parameter based URL's



Router JS

```
when('/ShowOrder/:orderId', {
      templateUrl: 'templates/show_order.html',
      controller: 'ShowOrderController'
});
```

Controller

```
$scope.order_id = $routeParams.orderId;
```

Services

- Services are singletons, which are objects that are instantiated only once per app (DI).
- They provide an interface to keep together methods that relate to a specific function.
- There is only one instance of a specific service available during the whole lifetime of the Angular application
- There are many internal services angular js provides
  Eg: $http, $route, $window, $timeout etc., (All angularjs internal services generally starts with $ sign)

How to Declare?

Services

```
var app = angular.module('app', []);
app.service('some-service', function(){…});
```

```
app.controller('some-controller',
        ['$scope', 'some-service'],
        function(scope, service){....}]);
```

Demo

Factory

- A factory is a simple function which allows you to add some logic before creating the object.
- It returns the created object

```
module.factory('factoryName', function() {

    var factory = {};

    factory.method1 = function() {
            //..
        }

    factory.method2 = function() {
            //..
        }

    return factory; //returns object
});
```

Factory vs Service

Service

- Constructor functions of the object which are instantiated with the **new** keyword.
- In other words new FunctionYouPassedToService()
- This object instance becomes the service object that AngularJS registers and injects later to other services / controllers if required.

Factory

- factories are functions that return the object.
- When declaring factoryName as an injectable argument you will be provided with the value that is returned by invoking the function reference passed to module.factory.
- Returns object

Service

```
module.service('MyService', function() {
    this.method1 = function() {
            //..
        }

    this.method2 = function() {
            //..
        }
});
```

Factory

```
module.factory('MyService', function() {

    var factory = {};

    factory.method1 = function() {
            //..
        }

    factory.method2 = function() {
            //..
        }

    return factory;
});
```

$scope - advanced (1)

- Used to Hold the data that we need to pass to the view.
- It is glued to view and controller.
- Api's
  - $watch to observe the model
  - $apply to propagate the change to the view

$scope - advanced (2)

- $rootScope
  - Only one root scope per app
  - Data can be passed between different controllers using $rootscope(if the controllers are the in the scope of the current $root)
  - Alternative method to communicate
    - $emit - when you want that $scope and all its parents and $rootScope to hear the event.
    - $broadcast - when you want to send the data $scope itself and its children.
    - $on - To catch all the messages that are communicated

Demo

Assignment - Download

Questions?

Thanks!